A Multi-Objective Evolutionary Approach to Class Disjointness Axiom Discovery

Thu Huong Nguyen, Andrea G. B. Tettamanzi

A Multi-Objective Evolutionary Approach to Class Disjointness Axiom Discovery

Thu Huong Nguyen
Université Côte d'Azur, Inria, CNRS, I3S
Nice, France
thu-huong.nguyen@univ-cotedazur.fr

Andrea G. B. Tettamanzi
Université Côte d'Azur, Inria, CNRS, I3S
Nice, France
andrea.tettamanzi@univ-cotedazur.fr

***Abstract*—The huge wealth of linked data available on the Web (also known as the Web of data), organized according to the standards of the Semantic Web, can be exploited to automatically discover new knowledge, expressed in the form of axioms, one of the essential components of ontologies. In order to overcome the limitations of existing methods for axiom discovery, we propose a two-objective grammar-based genetic programming approach that casts axiom discovery as a genetic programming problem involving the two independent criteria of axiom credibility and generality. We demonstrate the power of the proposed approach by applying it to the task of discovering class disjointness axioms involving complex class expression, a type of axioms that plays an important role in improving the quality of ontologies. We carry out experiments to determine the most appropriate parameter settings and we perform an empirical comparison of the proposed method with state-of-the-art methods proposed in the literature.**

***Index Terms*—Ontology Learning, OWL Axiom, Disjointness Axiom, Genetic Programming, Grammatical Evolution, Multi-Objective Optimization.**

I. Introduction

The growth of the Semantic Web also known as the Web of data, where the Linked Open Data (LOD) is a prominent representative opens up exciting opportunities for ontology learning. Due to heterogeneous semantic resources on the Web, ontological knowledge bases (KBs) may turn out to incomplete and noisy. Specifically, the incompleteness refers to the lack of information in ontology while the noise is relevant to the issues of invalid information. To enhance the quality of an ontology, the existence of axioms can be considered as the agents in pinpointing errors and inconsistencies in KBs. In ontology construction and knowledge base enrichment, the automatic asquistion of axiom is a central task which goes under the name of *axiom learning*. Like other types of axioms, class disjointness axioms are used to check the consistencies of the information contained in the ontologies or to deduce new information. For example, a reasoner will be able to deduce an error, i.e., a logical inconsistency of facts in the ontology, whenever the class *Fish* is associated to a resource related to the class *Planet*, if there is a constraint of disjointness between the two concepts *Fish* and *Planet*.

As a consequence of the essential role of class disjointness axioms in existing ontologies, learning hidden knowledge in terms of axioms from a LOD repository in the context of the Semantic Web has been the object of research using several different methods. Some prominent research towards the automatic creation of class disjointness axioms from RDF facts include supervised classification, like in the *LeDA* system [1], statistical schema induction via associative rule mining, like in the *GoldMiner* system [2], and learning general class descriptions (including disjointness) from training data, like in the DL-Learner framework [3]. In adtition, recent contribution has proposed using unsupervised statistical approaches like Formal Concept Analysis (FCA) [4] or Terminological Cluster Trees (TCT) [5], to discover disjointness axioms. Most approaches to learning axioms in the literature are based on deterministic level-wise generate-and-test methods, which essentially perform an exhaustive search, coupled with heuristic pruning, of the the space of hypotheses. Their main limitation is that they are incapable of scaling up when the space of the hypothesis, i.e. axioms, becomes too large. As a consequence, their applicability is restricted to the discovery of relatively simple axioms, i.e. atomic axioms.

Given the complexity of the problem, a heuristic approach, such as evolutionary algorithm, can handle with the search for more complex axioms whose space is incomparably larger. In fact, there are also some recent works [6]–[9] applying an evolutionary approach by using *Grammatical Evolution (GE)* to extracting class disjointness axioms from large RDF repository, i.e., DBpedia[1]. The use of a grammar allows great flexibility: only the grammar needs to be changed to mine different data repositories for different types of axioms. Extracted axioms in [6], [7] include both atomic and complex axioms, i.e., defined with the help of relational operators of intersection and union; in other words, axioms like $\mathsf{Dis}(C_1, C_2)$, where C_1 and C_2 are complex class expressions including operators. However, the dependence on SPARQL endpoints (i.e., query engines) for testing mined axioms against facts, i.e. instances, in large RDF repositories limits the performance of the method. In addition, evaluating the effectiveness of the method requires the participation of experts in specific domains, i.e. the use of a *Gold Standard*, which is proportional to the number of concepts. Hence, the extracted axioms are limited to the classes relevant to a small scope of topics, namely the *Work* topic of DBpedia. Also, complex axioms are defined with the help of relational operators of intersection and

[1]https://wiki.dbpedia.org/

union, which can also be mechanically derived from the known atomic axioms. To overcome that limitation, the type of mined class disjointness axioms in [8], [9] is extended to include the existence restriction ($\exists r.C$) and value restriction ($\forall r.C$) constructors, where r is a property and C a class, which cannot be mechanically derived from a given set of atomic axioms. Furthermore, a training-testing model is applied to objectively validate the method. Specifically, the whole DBpedia is used as the objective benchmark for evaluating the extracted axioms from a small training RDF dataset, i.e. sample of DBpedia, and eliminating the use of Gold Standard created by knowledge experts. The evaluation framework based on possibility theory [7]–[9] to determine the fitness values of generated axioms in the evolutionary cycle, i.e. the credibility and generality of axioms. However, the selection pressure in each phases of the evolutionary process tends naturally to drive the diversity of the population down. In addition, the presence of highly fit but possibly invalid candidate axioms in the population reduces the number of valid axioms that can be discovered which can be derived from unsuited fitness function in evaluation framework which based on a single criterion.

Along the lines of the studies using GE to mine class disjointness axioms, we extend the approach as a multi-objective problem, i.e. multi-objective GE, in addition to the trade-off between a set of objectives. Specifically, we used an multi-objectives approach to refine the evaluation of candidate axioms that improves the adaptive fit of a population of candidate axioms constrained by two independent criteria, i.e. the credibility and generality. We also proposed a new measure called *the similarity* and a method to compute it. We aim to optimize the evaluation framework for the axioms which ensures the high accuracy, generality and the diversity of the obtained axioms. In the study, we reuse the grammar and the training-testing model to extract class disjointness axioms in [8], [9] and perform comparison with the relevant approaches.

The rest of the paper is organized as follows: some basics in GE and GE in axioms discovery are provided in Section II. Axiom discovery in multi-objective GE approach is presented in Section III. Section IV introduces the organization of dataset. The experimental settings and results are presented and discussed in Section V. Finally, conclusions and directions for future research are given in Section VI.

II. Grammatical Evolution for Axiom Discovery

The foundation of our study is *Grammatical Evolution* (GE), a recent evolutionary model pioneered by Michael O'Neil and his collaborators [10]. In this section, we provide a summary of the GE in addition to theoretical and technical ingredients concerning axiom discovery.

A. Basics

Grammatical Evolution (GE) is an evolutionary approach that extends *Genetic Programming* (GP) [11], [12] to allow the exploration of the space of computer programs through the use of a grammar-mediated representation. Programs, viewed as *phenotypic solutions* or *phenotypes*, are decoded from variable-length binary strings, i.e., *genotypic individuals* or *genotypes*, through a transformation called *mapping process*. According to it, the variable-length binary string genomes, or *chromosomes*, are split into consecutive groups of bits, called *codons*, representing an integer value, used to select, at each step, one of a set of production rules from a formal grammar, typically in *Backus-Naur form (BNF)*, which specifies the syntax of the desired programs. Furthermore, inspired by biological evolution and its fundamental mechanisms, these programs are "bred" using iterative improvement of an initially random population of programs. That is an *evolutionary process*. At each iteration, known as a *generation*, improvements are made possible by stochastic variation, i.e., by a set of *genetic operators*, usually *crossover* and *mutation*, and probabilistic selection according to pre-specified criteria for judging the quality of an individual (solution). According to the levels of fitness, the process of selecting individuals, called *fitness-based selection*, is performed to create a list of better qualified individuals as input for generating a new set of candidate solutions in the next generation. The new solutions of each generation are bred by applying genetic operators on the selected old individuals. Then, *replacement* is the last step and decides which individuals stay in a population and which are replaced on a par, with selection influencing convergence. An illustration of the GE mechanism is presented in Fig 1.

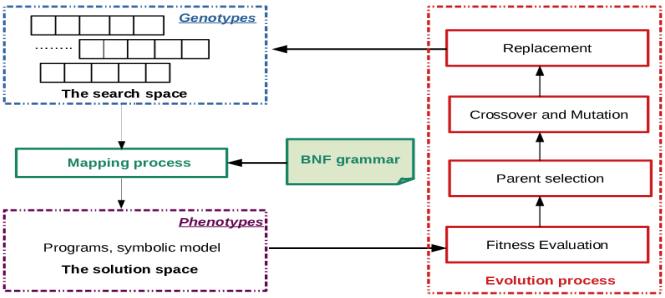

Fig. 1. Grammatical Evolution mechanism

B. BNF Grammar

In terms of axioms discovery, "programs" or "phenotypes" refer to axioms constrained by a given *BNF grammar*. A *BNF grammar* is a context-free grammar consisting of terminal and non-terminal symbols (also called just terminals and non-terminals) and being represented in the form of a four-tuple $\{N, T, P, S\}$, where N is the sets of non-terminals, which can be extended into one or more terminals; T is the set of terminals, which are tokens in the described language; P is the set of the production rules that map N to T; S is the start symbol and a member of N. When there are a number of productions that can be used to rewrite one specific non-terminal, they are separated by the '|' symbol.

We comply with the BNF grammar given in [8], [9] to mine binary disjointness axioms only, of the form $\mathsf{DisjointClasses}(C_1, C_2)$, where C_1 and C_2 may

be atomic or complex classes involving relational operators, i.e. existential quantification and value restriction, and possibly including more than one single class identifier, like DisjointClasses(VideoGame, ObjectAllValuesFrom(hasStadium, Sport)). To make the paper self-contained, we recall here the most important aspects of this grammar. The grammar is split into a static and a dynamic part to ensure that changes in the contents of RDF repositories will not require the grammar to be rewritten. The static part defines the syntax of the types of axioms to be extracted. The content of this part is loaded from a hand-crafted text file. The structure of the BNF grammar here aims at mining well-formed axioms expressing the facts, i.e. instances, contained in a given RDF triple store. Hence, only resources that actually occur in the RDF dataset should be generated. The static part of the grammar is thus structured as follows:

```
(r1) Axiom := ClassAxiom
(r2) ClassAxiom := DisjointClasses
(r3) DisjointClasses := 'DisjointClasses' '(' ClassExpression1 ' 'ClassExpression2 ')'
(r4) ClassExpression1 :=  Class               (0)
                        | ObjectSomeValuesFrom  (1)
                        | ObjectAllValuesFrom   (2)
                        | ObjectIntersection    (3)
(r5) ClassExpression2 :=  ObjectSomeValuesFrom  (0)
                        | ObjectAllValuesFrom   (1)
(r6) ObjectIntersectionOf := 'ObjectIntersectionOf' '(' Class ' ' Class ')'
(r7) ObjectSomeValuesFrom := 'ObjectSomeValuesFrom' '(' ObjectPropertyOf ' ' Class ')'
(r8) ObjectAllValuesFrom  := 'ObjectAllValuesFrom' '(' ObjectPropertyOf ' ' Class ')'
```

The dynamic part contains production rules for the low-level non-terminals, called *primitives* in [6], [7]. These production rules are automatically filled at run-time by querying the SPARQL endpoint of the RDF data source at hand. Let us consider an example representing a small sample of an RDF dataset:

```
PREFIX dbr: http://dbpedia.org/resource/
PREFIX dbo: http://dbpedia.org/ontology/
PREFIX dbprop: http://dbpedia.org/property/
PREFIX rdf: http://www.w3.org/1999/02/22\-rdf-syntax-ns\#

dbr:Amblycera       rdf:type            dbo:Animal.
dbr:Salweenia       rdf:type            dbo:Plant.
Dbr:Himalayas       rdf:type            dbo:NaturalPlace.
dbr:Amadeus         rdf:type            dbo:Work.
dbr:Cat_Napping     dbprop:director     dbr:William_Hanna.
dbr:With_Abandon    dbprop:artist       dbr:Chasing_Furies.
dbr:Idris_Muhammad  dbprop:occupation   dbr:Drummer.
dbr:Genes_Reunited  dbo:industry        dbr:Genealogy.
```

The productions for Class and ObjectPropertyOf would thus be:

```
(r9) Class :=  dbo:Animal       (0)
             |  dbo:Plant        (1)
             |  dbo:NaturalPlace (2)
             |  dbo:Work         (3)

(r10) ObjectPropertyOf :=  dbprop:director      (0)
                         |  dbprop:artist        (1)
                         |  dbprop:occuptation   (2)
                         |  dbo:industry         (3)
```

C. Mapping Process

In the *mapping process*, codons are used consecutively to choose production rules in the BNF grammar according to the function:

$$production = codon \ \mathbf{modulo} \left[\begin{smallmatrix}\text{Number of productions for the}\\ \text{current non-terminal}\end{smallmatrix}\right] \quad (1)$$

We illustrate the decoding of an integer chromosome into an OWL class disjointness axiom through a specific example (see Fig 2). Let the chromosome be (352, 265, 529, 927, 419). There is only one production for the non-terminals `Axiom`, `ClassAxiom`, `DisjointClasses`, `ObjectIntersectionOf`, `ObjectSomeValuesFrom` and `ObjectAllValuesFrom`, as it can be seen from Rules 1–3 and 6–8. In these cases, we skip using any codons for mapping and concentrate on reading codons for non-terminals having more than one production, like in Rules 4, 5, 9, and 10.

III. Multi-Objective GE for Axiom Discovery

Within the evolutionary process, the evaluation framework quantifies the quality of axioms, which is the base for selecting individuals (solutions) for the recombination, mutation, and replacement phases. In previous work [7]–[9], the aim was to look for credible and general axioms, based on possibilistic axiom scoring for credibility and on a scoring of their generality. These two scores were then combined into a single fitness function, i.e., a single objective. The superiority of an axiom over other ones was simply determined by comparing their fitness scores. However, the fitness values can suffer from one of two objectives offsetting the other, for instance when a high generality score is assigned to axioms possessing a low possibility. In fact, the two criteria of generality and credibility are incommensurable and any way of combining them is therefore largely a matter of points of view, hard to justify on an objective basis. To overcome that limitation, we treat the problem as a multi-objective optimization problem, which allows for explicit trade-offs with respect to a set of objectives.

A. Multi-Objective Evolutionary Algorithms

A *Multi-objective Optimization* (MO) [13] problem involves a number of objective functions constituting a multi-dimensional objective space, in addition to the decision variable space. Specifically, a solution to a MO problem is a vector of decision variables $x = (x_1, x_2, ..., x_n)^T$ in the decision space X. For each x, there exists an objective vector $y = (y_1, y_2, ..., y_n)^T$ in the objective space Y mapped by a function $f : X \to Y$.

The term *domination* is used for the situation of comparing two solutions $x^{(1)}$ and $x^{(2)}$. A solution $x^{(1)}$ dominates the other solution $x^{(2)}$ ($x^{(1)} \succ x^{(2)}$) if $x^{(1)}$ is no worse than $x^{(2)}$ in all objectives (no component of $y^{(1)}$ is smaller than the component of $y^{(2)}$, where $y^{(1)} = f(x^{(1)})$ and $y^{(2)} = f(x^{(2)})$), and $x^{(1)}$ is strictly better than $x^{(2)}$ in at least one objective (at least one component is greater). The set of optimal solutions in the decision space X is denoted as *Pareto-optimal solutions* or *Pareto set*. In addition, there are corresponding optimal objective vectors, i.e. points, in the objective space Y, denoted as *Pareto-optimal front* or *non-domination front*. In MO, all objectives are equally important, i.e., finding the optimum solution cannot be based on one objective alone while skipping other objectives. The goal of MO is to find multiple solutions

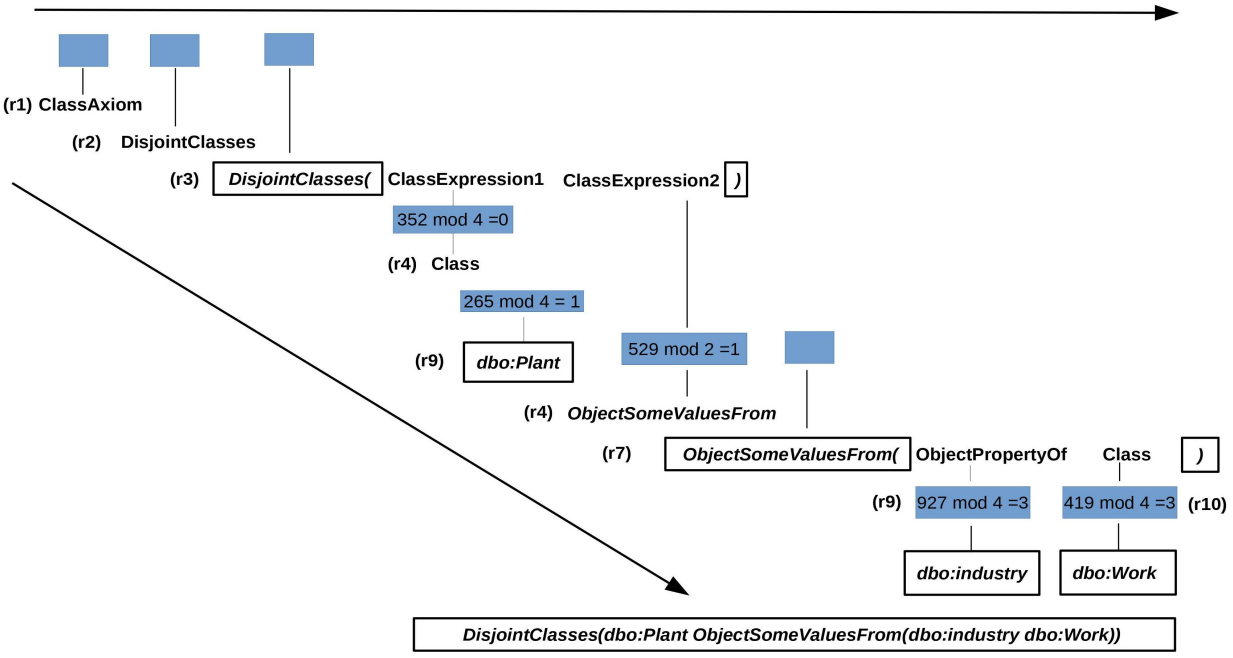

Fig. 2. An illustration of mapping process

representing the possible non-dominated trade-offs among the objective functions, i.e., a set of solutions lying on the Pareto-optimal front. In addition, a set of obtained solutions is sought for that is also diverse enough to represent the entire range of the Pareto-optimal front.

This results in a heuristic approach, the *Multi-objective Evolutionary Algorithm* (MOEA) [13], [14], which follows the goal of MO but refers to finding multiple non-dominated points as close to the Pareto-optimal front as possible, i.e., a *Pareto-optimal front approximation*, with respect to the trade-off among objectives. Also, it provides operators, i.e., recombination and mutation operators, to constantly improve the evolving non-dominated points.

NSGA-II [15] is one of the well-known multi objective evolutionary algorithms, which simultaneously optimizes each objective without being dominated by any other solution. NSGA-II concentrates on finding non-dominated solutions in addition to elistist and diversity preserving mechanisms.

As a particular case of MOEA, the approach we propose comprises the integration of GE in MOEA i.e., using NSGA-II, for axiom discovery, which we call Multi-Objective GE (MOGE). Basically, the mechanism of MOGE is quite similar to the one of a MOEA, except that we define multi-objective problems using integer arrays called *codons* as decision variables. The codons do not define axioms, i.e., the programs, themselves, but provide instructions for deriving axioms using the BNF grammar through the mapping process explained above.

B. Multi-objective Evaluation Framework

The goodness of an axioms is determined by its dominance, whereby it obtains a score on each objective which is not dominated by the corresponding score of another axiom. To derive such axioms, we extend the classic GE approach presented in [6]–[9] to MOGE. We also develop separate objective functions to evaluate the fitness of each axiom. In order to ensure the diversity of the obtained axioms, a scoring of the similarity of each axiom to the other axioms in the population (essentially, a local phenotypic crowding measure) is also considered in the evaluation framework. In this section, we first recall axiom the scoring, regarding possibility and generality applied in [7]–[9]. In addition, we introduce a new scoring, called *the similarity*. Finally, the objective functions for the method are presented.

1) Possibility Measure: is based on possibility theory [16], a mathematical theory of epistemic uncertainty. In the open-world, the knowledge base represented by RDF repository is incomplete. Additionally, as a results of the heterogeneous and collaborative character of the LOD, there exist some missing and erroneous facts (instances) in RDF datasets, i.e. noisy knowledge. Hence, adopting an axiom scoring heuristics based on possibility theory (see [17] for the theoretical background) is well-suited. Accordingly, a candidate axiom ϕ is viewed as a hypothesis that has to be tested against the evidence contained in an RDF dataset. Its *content* is defined as a finite set of logical consequences

$$\text{content}(\phi) = \{\psi : \phi \models \psi\}, \tag{2}$$

obtained through the instantiation of ϕ to the vocabulary of the RDF repository; every $\psi \in \text{content}(\phi)$ may be readily tested by means of a SPARQL `ASK` query. The *support* of axiom ϕ, u_ϕ, is defined as the cardinality of $\text{content}(\phi)$. The support, together with the number of confirmations u_ϕ^+ (i.e., the number of ψ for which the test is successful) and the

number of counterexamples u_ϕ^- (i.e., the number of ψ for which the test is unsuccessful), are used to compute a degree of *possibility* $\Pi(\phi)$ for axiom ϕ, defined, for $u(\phi) > 0$, as

$$\Pi(\phi) = 1 - \sqrt{1 - \left(\frac{u_\phi - u_\phi^-}{u_\phi}\right)^2}.$$

Possibility alone is a reliable measure of the *credibility* of a class disjointness axiom, all the more so because (and this is a very important point), in view of the open world assumption, for two classes that do not share any instance, disjointness *can only be hypothetical* (i.e., fully possible, if not contradicted by facts, but *never* necessary). Possibility is measured by defining the numbers of counterexamples and the support. These values are counted by executing the corresponding SPARQL queries based on *graph patterns*, via an accessible SPARQL endpoint. We refer the interested reader to [7], [8] for an in-depth description of the relevant SPARQL queries.

2) Generality measure: is determined by the quantities of the facts(instances), in the extension of its components. In [6], the generality of an axiom is defined as the cardinality of the set of the facts in the RDF repository reflecting the support of each axiom, i.e., u_ϕ. However, in case one of the components of an axiom is not supported by any fact, its generality should be zero. Hence, the generality of an axiom should be measured by *the minimum* of the cardinalities of the extensions of the two class expressions involved, i.e. $g_\phi = \min\{\|[C]\|, \|[D]\|\}$ where C, D are class expressions.

3) Similarity measure: quantifies the similarity of an axiom ϕ, $s(\phi)$, to the population of n axioms which is defined by the average of similarity metrics $s(\phi, a_i)$ between axiom ϕ and each axiom a_i in the population:

$$s(\phi) = \frac{1}{n-1} \sum_{i=1; a_i \neq \phi}^{n} s(\phi, a_i) \tag{3}$$

In order to measure the similarity coefficient $s(\phi)$ as in the above formula, the similarities $s(\phi, a_i)$ need to be computed. As mentioned in II-B, axioms are structured in the form of binary axioms of the form $\phi \equiv DisjointClasses(A, B)$ and $a_i \equiv DisjointClasses(C, D)$ where A, B, C, D can be atomic expressions or complex expressions containing relational operators of restriction, i.e., existential quantification and value restriction. We define the similarity between two axioms based on the similarities between pairs of expressions as

$$s(\phi, a_i) = \max\{s(A, C), s(A, D), s(B, C), s(B, D)\} \tag{4}$$

Expressions in each axiom are represented in the form of binary trees where each node can be an atomic class or a relational operator, namely existential quantification ($\exists$), value restriction ($\forall$), or intersection ($\sqcap$) operators. Determining each similarity between expressions, e.g., $s(A, C)$, is performed on corresponding binary trees t_1 and t_2. Binary trees are traversed simultaneously and each pair of corresponding nodes (p_j, q_j) in both trees, i.e., p_j in t_1 and q_j in t_2, is compared to each other and the value returned is the similarity between two nodes, i.e., $s(p_j, q_j)$, according to Table I. One notable point is that if both nodes represent atomic classes, the value returned is 1 if the two nodes represent the same class; otherwise the value returned is 0. Each similarity between expressions, e.g. s(A,C), is defined as

$$s(A, C) = \frac{1}{k} \sum_{j=1}^{k} s(p_j, q_j) \tag{5}$$

where k is the number of pairs defined by the number of nodes in the smallest tree.

TABLE I
MATRIX FOR THE COMPARISON BETWEEN NODES

Node	Atomic class	$\sqcap$	$\exists$	$\forall$
Atomic class	0 or 1	0	0	0
$\sqcap$	0	1	0	0.5
$\exists$	0	0	1	0
$\forall$	0	0.5	0	1

4) Objective Functions: are used for the comparisons between axioms which reflect the correlation of measures to determine the quality of each axiom. We propose two objective functions, f_1 and f_2, used in our approach, which aim at obtaining axioms that maximize the value of possibility and generality, while not being too similar among themselves:

$$\begin{cases} \text{Maximize } f_1 = \Pi(\phi) \cdot \sqrt{1 - s(\phi)^2} \\ \text{Maximize } f_2 = g_\phi \cdot \sqrt{1 - s(\phi)^2} \\ \text{Where } 0 \leq \Pi(\phi) \leq 1;\ g_\phi \geq 0\ ;\ 0 < s(\phi) < 1 \end{cases} \tag{6}$$

IV. DATASET ORGANIZATION

To investigate the effectiveness of our approach, we organize our dataset following the "training-testing" model. Specifically, the learning process is performed with the input data source derived from a training RDF dataset, a random sample of DBpedia, whereas the evaluation of discovered axioms is based on a testing dataset, which is the full DBpedia, which can be considered as an objective benchmark. The workflow of this model is shown in Fig 3.

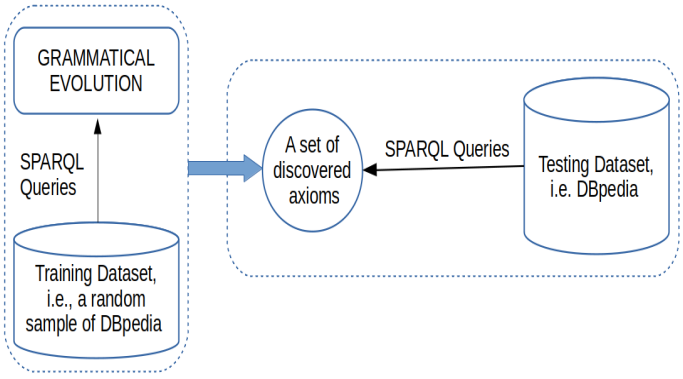

Fig. 3. Workflow of class disjointness axioms discovery using GE in the training- testing model

We use the same *Training Dataset*[2] *(TD)* used in [8], [9], with 6,739,240 connected RDF triples with a variety of topics from DBpedia, which randomly collect 1% of the triples from DBpedia 2015-04 (English version).The performance of the method is measured by using the entire DBpedia 2015-04 as a test set, measuring possibility, generality, and similarity scores for every distinct axiom discovered by our algorithm. To avoid overloading DBpedia's SPARQL endpoint, we set up a local mirror using the Virtuoso Universal Server.[3]

V. Experiments & Results

A. Experimental Protocol

We use the BNF grammar introduced in Section II. In addition, to make fair comparisons possible with previous studies [8], [9], a set of milestones of total effort k (defined as the total number of fitness evaluations) corresponding to each population size are also recorded for each run, namely 100,000; 200,000; 300,000 and 400,000, respectively. The maximum numbers of generations, *maxGenerations* (used as the stopping criterion of the algorithm) are automatically determined based on the values of the total effort k, thus $popSize \cdot maxGenerations = k$.

A prototype system of the proposed method was developed in Java, using Apache Jena to interface with SPARQL endpoints and GEVA v.2.0 [4], a reference Java implementation of GE. Also, we integrated the system with the MOEA framework API,[5] a Java framework for multi-objective optimization. The parameters are listed in Table II.

TABLE II
Parameter values for MOGE.

Parameter	Value
Total effort k	100,000; 200,000; 300,000; 400,000
initLenChrom	6
pCross	80%
popSize	1000; 2000; 5000; 10000

B. Results

We ran the *MOGE* method for 20 distinct runs for each of the different parameter settings summarized in Table II, using the BNF grammar defined in Section II-B. The full set of valid distinct axioms, i.e., axioms ϕ such that $\Pi(\phi) > 0$ and $g_\phi > 0$ discovered are available online.[6] Statistics for automatically generated axioms are presented in Table III. In addition, we can see in Fig. 4 that the number of valid distinct axioms for most parameter settings, i.e., population size *popSize* and total effort *k*, mined by *MOGE* is significantly greater than those mined by the single-objective GE method in [8], [9]. This means that the diversity of an extracted set of axioms is considerably enhanced when we use the *MOGE* method.

[2]Available for download at http://bit.ly/2OtFqHp
[3]https://virtuoso.openlinksw.com/
[4]http://ncra.ucd.ie/Site/GEVA.html
[5]http://moeaframework.org/javadoc/index.html
[6]https://bit.ly/38crj4M

TABLE III
Number of valid distinct axioms discovered over 20 runs

k \ popSize	1000	2000	5000	10000
100000	8084	16085	41320	50535
200000	8713	17400	41813	76804
300000	7970	17680	40303	70562
400000	8457	16258	40656	67722

Furthermore, we follow the use of the fuzzy extension of the usual definition of *precision* in [8], [9] to measure the accuracy of our results. Accordingly, $\Pi(\phi)$ is interpreted as the degree of membership of axiom ϕ in the (fuzzy) set of the "positive" axioms. The value of precision can thus be computed against the test dataset, i.e., DBpedia 2015-04, according to the formula

$$\text{precision} = \frac{\|\text{true positives}\|}{\|\text{discovered axioms}\|} = \frac{\sum_\phi \Pi_{\text{DBpedia}}(\phi)}{\sum_\phi \Pi_{\text{TD}}(\phi)}. \quad (7)$$

where Π_{TD} and Π_{DBpedia} are the possibility measures computed on the training dataset and DBpedia, respectively.

The results in Table IV confirm the high accuracy of the proposed *MOGE* method. The precision values are quite equivalent to the figures of *GE* method [8], [9] with the range from 0.984 to 0.996 for all the different considered population sizes and different numbers of generations (reflected through the values of total effort).

Fig. 5 illustrates the distribution of axioms having $\Pi(\phi) > \frac{2}{3}$ in terms of the two objectives, i.e. possibility and generality, compared with the *GE* methods of [8], [9]. We perform the comparison based on the results of the best setting, i.e., those yielding the largest number of obtained distinct axioms and the highest accuracy, for either method, i.e., $\{popSize = 10,000; k = 200,000\}$ and $\{popSize = 5,000; k = 300,000\}$, respectively. We can observe that the number of highly qualified axioms ($\Pi(\Phi) > \frac{2}{3}$ and $g_\Phi > 100$)) is maintained in *MOGE* system. More clearly, based on the specific resulting statistics, the number of obtained axioms from MOGE in the best setting is $38,134$ which is much greater than those extracted by the GE, i.e., $23,767$ axioms. In addition, with the smaller value of total effort k reflecting the cost of evaluations, i.e., $k = 200,0000$ compared with $k = 300,000$ in [8], [9], *MOGE* is clearly more effective in inducing highly qualified axioms. We also show the distribution of the discovered axioms in this best setting in terms of similarity coefficient in Fig. 6. The range of similarity scores recorded for these axioms lies below 0.35, which indicates a good diversity of the classes and properties in the components of axioms. Based on the given grammar, one part of the axioms is forced to contain a relational operator, i.e. $\exists$, $\forall$, or $\sqcap$, hence, the overlap of the operators in axioms does not allow the similarity score to be zero.

According to the results, we consider in detail the axioms discovered by the algorithm with this best setting. First, we witness that the number of obtained axioms containing the $\exists$ operator is slightly larger than the one

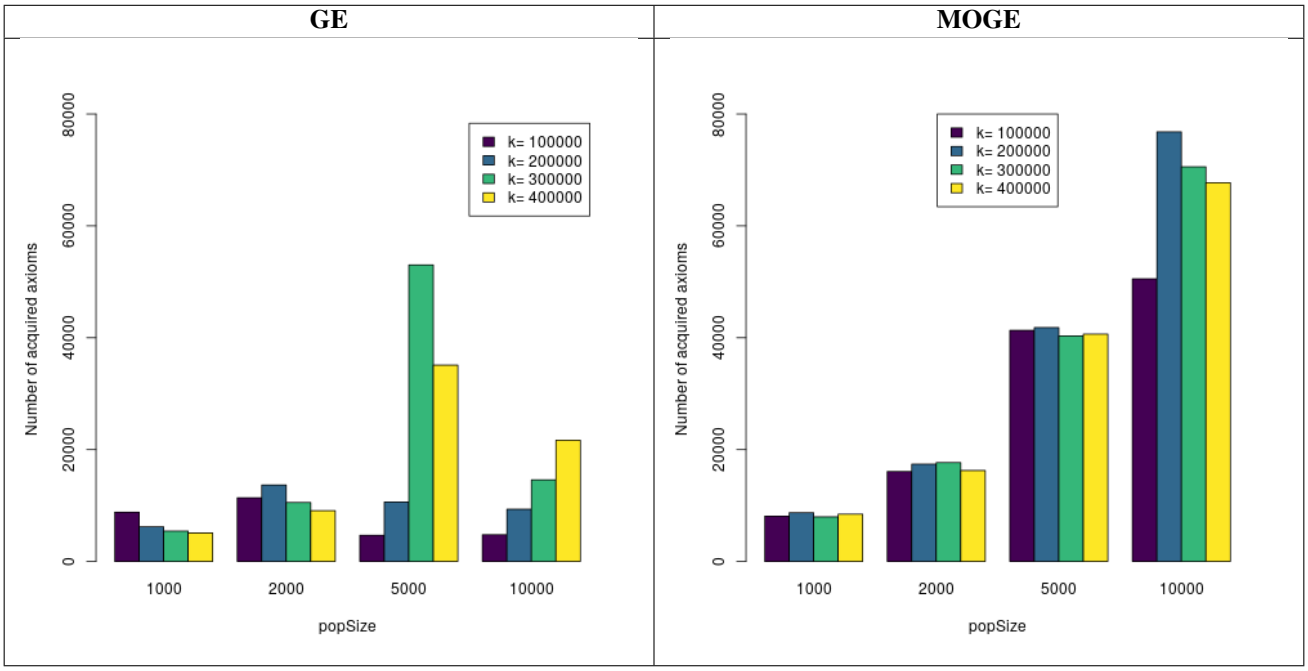

Fig. 4. Statistical comparison about the number of axioms discovered over 20 runs between MOGE and GE method.

TABLE IV
AVERAGE PRECISION PER RUN ($\pm$***std***)

	GE				MOGE			
k \ *popSize*	1,000	2,000	5,000	10,000	1,000	2,000	5,000	10,000
100,000	0.981 0.019	0.999 $\pm$0.002	0.998 $\pm$0.002	0.998 $\pm$0.003	0.988 $\pm$0.007	0.990 $\pm$0.005	0.989 $\pm$0.003	0.996 $\pm$0.001
200,000	0.973 $\pm$0.024	0.979 $\pm$0.011	0.998 $\pm$0.001	0.998 $\pm$0.002	0.989 $\pm$0.007	0.990 $\pm$0.004	0.987 $\pm$0.004	0.988 $\pm$0.004
300,000	0.972 $\pm$0.024	0.973 $\pm$0.014	0.993 $\pm$0.007	0.998 $\pm$0.001	0.989 $\pm$0.007	0.989 $\pm$0.003	0.986 $\pm$0.004	0.986 $\pm$0.003
400,000	0.972 $\pm$0.024	0.969 $\pm$0.018	0.980 $\pm$0.008	0.998 $\pm$0.001	0.989 $\pm$0.008	0.990 $\pm$0.003	0.985 $\pm$0.004	0.984 $\pm$0.004

of those with the $\forall$ operator, namely 40,122 and 36,682 axioms, respectively. However, together with the mandatory class expression containing the $\forall$ or $\exists$ operator, most extracted class disjointness axioms contain an atomic class expression. This may be due to the fact that the support of atomic classes is usually larger than the support of a complex class expression. Specifically, we obtain 7 axioms containing complex expressions in both their members. These axioms are less general, even though they are completely possible. An example is the case with DisjointClasses(ObjectAllValuesFrom(dbprop:operation dbo:MilitaryConflict) ObjectAllValuesFrom(dbprop:order dbo:MIlitaryUnit))($\Pi(\phi) = 1.0$; $g_\phi = 1$). Also, we analyze an example of a completely possible and highly general axiom, DisjointClasses(dbo:District ObjectSomeValuesFrom(dbo:birthPlace dbo:Place)) ($\Pi(\phi) = 1.0$; $g_\phi = 8,483$), which we can paraphrase as "*districts cannot have a place as their birthplace*". Knowing that District and Place are *not* disjoint, this axiom states that District and $\exists$birthPlace.Place are in fact disjoint; in addition, $\exists$birthPlace.Place, i.e., "(people) whose birthplace is a place" is a class with many instances, hence the high generality of the axiom.

VI. CONCLUSION

We have proposed a multi-objective extension to a grammar-based genetic programming approach to axiom discovery which consists of using two objectives plus a "similarity" score, which is in fact a sort of a local phenotypic crowding factor. The experimental results confirm that the proposed method is capable of discovering highly accurate and general axioms and is more effective compared with the single-objective methods of previous studies. In the future, we will focus on mining disjointness axioms involving further types of complex classes, by bringing into the picture other relational operators such as `owl:hasValue` and `owl:OneOf`. We also plan on refining the evaluation of candidate axioms with the inclusion of some measurement of their complexity in the fitness function.

ACKNOWLEDGMENTS

This research was carried out in the Wimmics team, which is a joint research team of Université Côte d'Azur, Inria, and

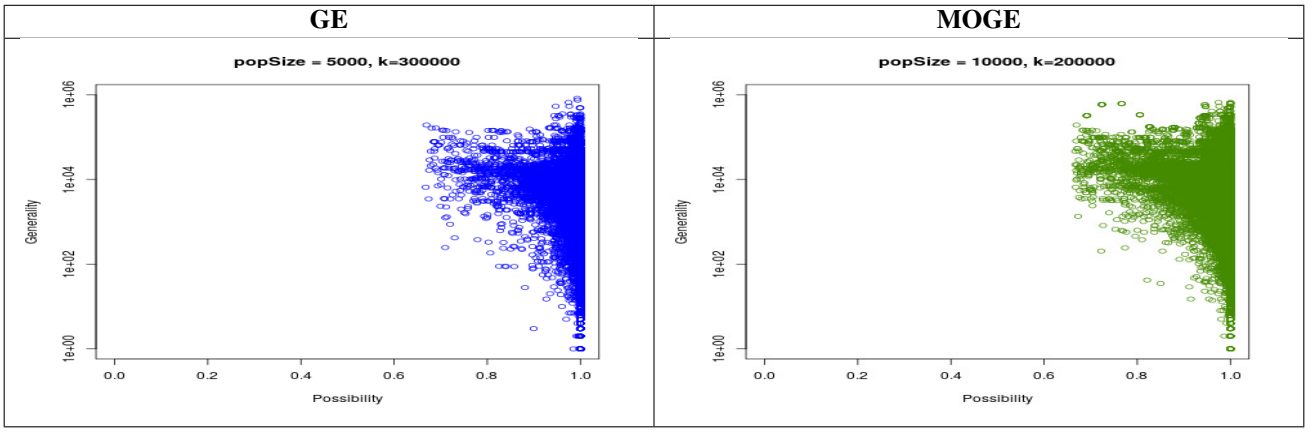

Fig. 5. Possibility and generality distribution of the discovered axioms with $\Pi(\phi) > \frac{2}{3}$

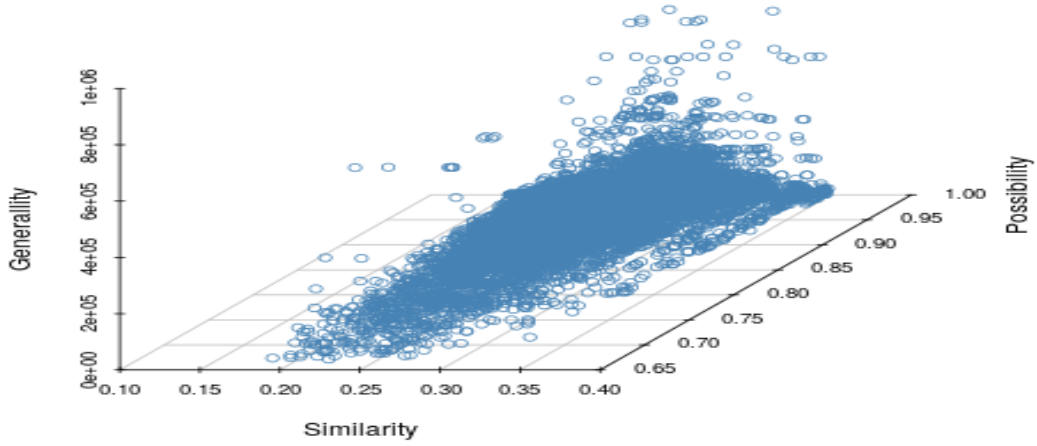

Fig. 6. The distribution of the discovered axioms in terms of measures ($\Pi(\phi) > \frac{2}{3}$)

I3S. Our research motto: AI in bridging social semantics and formal semantics on the Web.

This work has been partially supported by the French government, through the 3IA Côte d'Azur "Investments in the Future" project managed by the National Research Agency (ANR) with the reference number ANR-19-P3IA-0002.

REFERENCES

[1] J. Völker, D. Vrandecic, Y. Sure, and A. Hotho, "Learning disjointness," in *ESWC*, ser. Lecture Notes in Computer Science, vol. 4519. Springer, 2007, pp. 175–189.

[2] J. Völker, D. Fleischhacker, and H. Stuckenschmidt, "Automatic acquisition of class disjointness," *J. Web Sem.*, vol. 35, pp. 124–139, 2015.

[3] J. Lehmann, "Dl-learner: Learning concepts in description logics," *Journal of Machine Learning Research*, vol. 10, pp. 2639–2642, 2009.

[4] J. Reynaud, Y. Toussaint, and A. Napoli, "Redescription mining for learning definitions and disjointness axioms in linked open data," in *ICCS*, ser. Lecture Notes in Computer Science, vol. 11530. Springer, 2019, pp. 175–189.

[5] G. Rizzo, C. d'Amato, N. Fanizzi, and F. Esposito, "Terminological cluster trees for disjointness axiom discovery," in *ESWC (1)*, ser. Lecture Notes in Computer Science, vol. 10249, 2017, pp. 184–201.

[6] T. H. Nguyen and A. G. B. Tettamanzi, "Learning class disjointness axioms using grammatical evolution," in *EuroGP*, ser. Lecture Notes in Computer Science, vol. 11451. Springer, 2019, pp. 278–294.

[7] ——, "An evolutionary approach to class disjointness axiom discovery," in *WI*. ACM, 2019, pp. 68–75.

[8] ——, "Using grammar-based genetic programming for mining disjointness axioms involving complex class expressions," in *ICCS*. Springer, 2020.

[9] ——, "Grammatical evolution to mine OWL disjointness axioms involving complex concept expressions," in *WCCI*. IEEE, 2020.

[10] M. O'Neill and C. Ryan, "Grammatical evolution," *Trans. Evol. Comp*, vol. 5, no. 4, pp. 349–358, Aug. 2001. [Online]. Available: http://dx.doi.org/10.1109/4235.942529

[11] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[12] L. Vanneschi and R. Poli, *Genetic Programming — Introduction, Applications, Theory and Open Issues*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012.

[13] K. Deb, "Multi-objective optimisation using evolutionary algorithms: An introduction," in *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*. Springer, 2011, pp. 3–34.

[14] E. Zitzler, M. Laumanns, and S. Bleuler, "A tutorial on evolutionary multiobjective optimization," in *Metaheuristics for Multiobjective Optimisation*, X. Gandibleux, M. Sevaux, K. Sörensen, and V. T'kindt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 3–37.

[15] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.

[16] L. A. Zadeh, "Fuzzy sets as a basis for a theory of possibility," *Fuzzy Sets and Systems*, vol. 1, pp. 3–28, 1978.

[17] A. G. B. Tettamanzi, C. Faron-Zucker, and F. Gandon, "Possibilistic testing of OWL axioms against RDF data," *Int. J. Approx. Reasoning*, vol. 91, pp. 114–130, 2017.